Defendants claimed that they "did not preserve" any of the Seed Selection Code for LDS64 and LDS100, and that the Production contained no sequence development files or testing files for LDS64 and LDS100. We will see later they also destroyed all original development files and testing files for LDS200.

88. Defendants' choice of the words that these code files were "not preserved" is misleading. To allow the small code written by Dr. Teytel that comprises the Seed Selection Code remain in his home directory in their central sever takes no effort. All together, it was less than 100 KB's as shown by their LDS200 files produced. The Development Production comprises 11 GB, most of which is useless purported outputs that Teytel claimed that he preserved for over 5 years from 2000 to 2005 in his home directory for the litigation — and then destroyed during the litigation. The Development Production is 100,000 times bigger in terms of file size than the destroyed relevant Seed Selection Code. In the directory lds200/logs2 from CGM7077 alone contains one Gigabytes of log files that are useless. Why would Defendants delete the key Seed Selection Code which occupied very little space while at the same time saving garbage code that takes up 100,000 times more space? There is no good reason to do that other than that Defendants sought to hide information that would be harmful to them if disclosed.

89. Dr. Teytel supposedly worked from 1998 through 1999 to write this code, continuously used it for LDS64 and then for LDS100, and was still using it for LDS200 development through 2000. So why would he destroy the code? Defendants purportedly saved all obsolete code in RCS archive files for many years. There is no reason to destroy those sequence selection code files except to cover up the fact the Real Algorithm targets ACE. This conclusion is supported by my 2nd Declaration ¶¶ 56-60 (Ex. E) which demonstrated that that the Mixed Seed Algorithm had no mathematical validity for very specific reasons. .

90. It cost nothing to let this relatively small amount code remain in the company's central server. It requires some work to delete or destroy files in a computer, so the intentional destruction must serve some purpose. Defendants purportedly save all obsolete code in RCS for many years. Dr. Teytel packed 29 identical copies of RCS files in the Development Production. Dr. Teytel provided no plausible explanation why did he destroyed all the relevant code files leaving only irrelevant and unverifiable files. The only logical reason to do so is to cover up the fact the Real Algorithm targets ACEand supports the conclusion that the Development Production was phony and simply part of the overall attempt to conceal their theft and use of ACE in furtherance of their arbitrage strategy that was enabled by ACE.

D2. Defendants destroyed all code files relating to sequence selection, <u>and</u> also variance reduction and testing for LDS100, many of which, as shown by Teytel's Notebook, have "ACE" in their names

91. In fact, Defendants didn't delete all of LDS64 development files as they claimed. CGM7079 contains a directory REDACTED for LDS64 was preserved, with their last modification dates are in Jan-Feb 1999. If they could preserve those files certainly they could be able to preserve all other code for LDS64. It is

interesting to see why they kept this part of code. Teytel testified that the REDACTED . Teytel surgically deleted the incriminating Seed Selection Code and left the REDACTED to survive his destruction.

92. Why did Teytel also destroy all the code files for REDACTED for LDS100? These files would not reveal the incriminating Algorithm. The answer was provided by Teytel's Notebook, which mentions "ACE" as many as 12 times in names of code files used relating to the testing of LDS100. Plainly, Teytel was testing ACE against various REDACTED of LDS100.

93. I have shown in my first Expert Report and 2nd Declaration (Ex. G and E) that the references to "ACE" and ACE related code in the Teytel Notebook (Ex. SS) indicated the use of ACE as a target for creating derivative sequences, not just ACE test results as claimed by Dr. Teytel . Below are some examples of the ACE and ACE related code in the Notebook and their meaning:

First, "ACE64 Comparison for lds 100" (p. CGM00238 in Ex. SS). The purpose of the comparison was to compare the accuracy of the targeted ACE64 with a derived 100 path mixed seed sequence.

Second, "gauss_random.c with fixed long seeding" (p. CGM00238 in Ex. SS). As explained in my 2nd Declaration (Ex. E, ¶ 44-45), Defendants did not have seeds to generate ACE sequences. Thus to use ACE, they had to take the ACE sequence numbers themselves which had to be stored in a data file and read them into The Yield Book to generate interest rate paths. They could not simply feed seeds into their random number generator. To accomplish this, Dr. Wang helped Robert Russell modify gauss_random.c which had as its default seed the integer – REDACTED See Ex. PPPP p.2. In C code terminology, the language that this code file was written in, term "short integers" means an integer with a maximum value of REDACTED LDS100 was generated by a "short integer". All the seeds used to generate it had an integer value of less than REDACTED See Ex. OOOO, p. 2. Thus the Teytel Notebook reference to "gauss_random.c with fixed long seeding" was a reference to the gauss_random.c modified to use the ACE sequences during the testing of ACE because that version of gauss_random.c had a "long" seed, a number greater than REDACTED as shown above. It means that the Notebook reference to "ACE64 Comparison for lds 100" means exactly what it says on its face, that Teytel was doing a comparison of the ACE64 sequence to lds100 using the ACE64 sequence itself, not just ACE test results.

Third, "cmoopt.ace" and "cmoprog.ace" (p. CGM00239 in Ex. SS). During testing, Teytel used an executable program "cmoopt" which generates a sequence using seeds being tested to calculate the value of the MBS bonds using the sequence. p.7, Second Expert Report of Nathaniel Polish, Ex. NN. The Notebook

[24] Teytel August 23, 2012 dep.; 101:4-5 "Yes. So from that perspective, it's an independent project."

clearly references "cmoopt" on that page of the Notebook. On the same page is the reference to "cmoopt.simple" which Teytel admitted was a reference to testing using the cmoopt program using the single seed Production200 sequence in production at the time to calculate the value of MBS bonds. So logically, the reference to "cmoopt.ace" to calculate the value of bonds is a reference to the program using the ACE sequence to calculate the value of bonds, especially since as, Dr. Polish noted, he was using "cmoopt" to test his mixed seed sequences. Such program would use gauss_random.c with fixed long seeding as explained above. "cmoprog.ACE is simply a script file that is read directly by the computer and calls in "cmoopt.ace" which automatically reads in the ACE sequence and the list of MBS bonds to be valued. See my first Expert Report (Ex. G, p. 14).

Other references to "ACE" in the Notebook are clearly also references to the use of ACE. The reference to LDS-ACE2 on CGM 240 indicates that these sequences used the same scenarios, the same Yield Curve and volatility information with the sequence to generate interest rate paths. On CGM 244, there appears "LDS-ACE2", "same as lds-ace, but different opt-scen!" (opt-scen means option scenario). Here they tried out different Yield Curves and volatility parameters when comparing LDS100 to ACE to see how closely it approximated ACE valuation of MBS.

94. All the Teytel development Notebook files with ACE in their names have been deleted by the Defendants. This is no coincidence. These Notebook references alone show that Teytel had the ACE64 sequence and was testing it against LDS100. These references confirm all the other evidence described in my previous Expert Reports and declarations. The Theft Code Shows how they stole it and that they stole all the ACE sequences.

D3. LDS200 files fabrication I: Teytel modified the metadata "Date Modified" of all LDS200 files

95. I have examined Defendants' production of sequence development and testing files, in 5 CD's bates stamped CGM07077-81 produced on or about March 8, 2011 ("Sequence Development File and Testing File Production" or the "Production"), and concluded that, they have been fabricated or otherwise heavily tampered with.

96. The vast majority of Production is contained in the four folders entitled "LDS200", "LDS200_1", "LDS200_2", and "LDS200_3", which comprise four out of the five CD's (CGM07077-80). Clearly, these files are all purportedly dedicated to the selection and testing of LDS200. However, all of these LDS200 files have "last modification date" in July 2005, five years after the LDS200 development and testing were completed according to Defendants' interrogatory responses (see Ex. W, Responses to Interrogatories 1 and 2).

97. Dr. Teytel could not explain what exactly he did when he purportedly copied these files in the Production in a way that changed all the "Date modified" dates of all the purported LDS200 files to July 2005. He suggested that he may have used command "tar" to compress the files and that he skipped a step which, for a reason he could not explain, caused modification of the "Date Modified". Teytel August 23, 2012 dep. 91:8-91:16, 91:16-93:17 (Ex. TT). It could

not have happened following the procedure he described, not when the files in CGM07081 were copied by him from his home directory without modifying the "Data Modified". The computer command "tar" does not change the last modification dates of the files, unless Dr. Teytel deliberately added the extra command "-m" to "tar", where the character "m" stands for "modification" [of last modification dates of the files]. So if Dr. Teytel created, or modified, any or all these files in July 2005 for discovery, that metadata modification command "tar –m" has erased all the traces of tampering.

D4. Development Production fabrication II: Teytel destroyed all of the original sequence development files and testing files to cover up the fabrication

98. Dr. Teytel testified that he has deleted from his home directory on Defendants' server the entire original files from which the Development Production was copied so no one can verify his claim that he made no changes in the files produced. Teytel August 23, 2012 dep. 95:8-16, 96:5-8, Ex. TT. Teytel testified that copies of his home directory were made by a "special archiving" group in 2001. Teytel August 23, 2012 dep. 48:21-49:9, (Ex. TT). Plaintiff requested immediately the production of these files. Defendants failed to produce them and provided no explanation for their original non-production.

99. The only files that have anything to do with the default seeds of LDS200 in the Production are contained in the handful files with names containing **REDACTED** Dr. Teytel's deletion made it nearly impossible to detect whether those few **REDACTED** were slipped into the Production in July 2005, years after the purported testing was completed. Defendants withheld the critical source code of The Yield Book and RCS code that are necessary to build any The Yield Book applications, including **REDACTED** that would have been useful for detection.

100. In ¶ 44 of the Fan Notice, I identified eleven head files from The Yield Book code and RCS that are missing, see Ex. A. Anyone who wants to build the cmoopt application will need these files, and many other code files called by these files, and the missing MR-RCS,TYB-RCS, the entire CMO-RCS code that have been withheld by Defendants. Defendants produced only one machine executable cmoopt without providing the source code to build it. Humans cannot read cmoopt. No one can determine what sequences were used by cmoopt to generate the large quantities of price outputs that constitute the bulk of Production, or even run cmoopt to verify they are outputs of cmoopt. No one run any script files upstream to cmoopt, like cmoprog.lib. Thus, no one can verify if any of the purported outputs were ever used by the script perl files or in any way by the Mixed Seed Algorithm of Teytel.

101. While Dr. Polish contended that believed that these files reflected the development of LDS200 as described in Teytel's Notebook in his June 18, 2012 Report (Ex. NN), he testified that he had never actually even run the application cmoopt, much less compiled and built a cmoopt, nor ever verified any of the outputs of cmoopt, see Polish July 16, 2012 Dep. 116:8-19, Ex. XX. Nor could he verify the Development Production from the files produced to Plaintiff. When purported data, the outputs and the seeds, like this are unreliable and unverifiable, any analysis based open these data are meaningless. No one can be sure a seed is real or that it was selected in a purported range, say 1—10000, or was selected based on a sorting

of some unverifiable unknown purported output files or based on its proximity to a certain ACE sequence.

D5. LDS200 fabrication III: Teytel packed the Production with tens of thousands of redundant RCS files

102. LDS200 development folders on DVD CGM07079 have names like REDACTED indicating they were the development folders for the LDS200. In those LDS200 development folders, I found a large number of new files and redundant files created in July 2005. In the LDS200_1 folder alone, RE RCS folders have been found packed into its REDAC different subfolders. Another REDACT RCS folders are compressed. Each of the REDAC RCS folders contains REDACTED of redundant code. I found that the files in each of the RE RCS folders are identical to the files in each of the other REDAC RCS folders. As indicated by the log files, each of the RE RCS folders was created in 2005. The "Date modified" stamp on each of these files indicates that these REDAC redundant, identical RCS directories were copied serially, one download after another, non-stop, into the REDA different subdirectories, starting at noon and finishing at 1 pm, approximately, on July 11, 2005.

103. I have been informed that Defendants refused to produce an updated RCS files post-2005, and refused to produce a single copy of The Yield Book, Inc.'s RCS files. If Defendants wanted to produce the partial RCS files from Mortgage Research in 2005, they could have produced a single copy of RCS files; they had already done so in CGM176 and 177. Even if these identical folders and files were created in 2000 rather than 2005, I cannot see any reason any programmer would want to create REDACT identical folders in his development folders, or cause the creation of the RE identical folders to confuse himself, only in the Production to confuse the opposing party. These large numbers of identical and redundant folders and files did cause confusion and difficulty for my analysis.

104. The REDACT identical RCS directories have been created in the folder REDACTED in disk CGM07079. These RCS directories had no utilities for the folders they reside in. The acronym REDACTED in REDACTED suggest the REDACTED of The Yield Book REDACTED for MBS like pass-throughs, IO's and PO's REDACTED or ARM's, name adjustable rates, and REDACTED for CMO's. Hence REDACTED The acronym REDACTED .

105. To build The Yield Book for testing, the source code was extracted from the REDACTED complete RCS files stored in the server, and stored in the subfolder REDACTED, namely

REDACTED

According to the library or the applications or the category of the source code extracted from RCS, each file is stored in one of the subfolders of REDACTED. Thus, there are RE subfolders of REDACTED in REDACTED .

106. Then appropriate "makefile" is then run to build the library or the applications. For example, REDACTED

REDACTED Clearly, there is no REDACTED reason to copy the RCS folder from the server into the REDACTED to build the REDACTED Dr. Teytel had no clear explanation for this, he said he did not know for sure, but REDACTED Teyel August 23, 2102 Dep. 114:9-14. This explanation is contradicted by the fact that none of the REDAC subfolders in another directory created to perform the identical tasks, REDACTED REDACT on CGM07081, has these RCS subfolders. REDACTED on CGM07081 has REDAC subdirectories with the identical names as REDACT out of REDAC subfolders names in the tampered with folder REDACTED on disk CGM07079. REDACTED on CGM07081 is the only disk which has the original latest modified dates from February 1999 when the development actually tood place. REDACTED on CGM07081, contains no RCS folder and no RCS files because there is no reason to have them there. If anyone wanted a link to the RCS folder, he would create just one link, not REDAC links. If at the time of production in for discovery Dr. Teytel deliberately copied the REDAC RCS links for the purposes of making analysis more difficult, the production would look exactly like what was produced. Because he destroyed the original files and modified all the dates, we cannot know for sure what was done. But the contradictory evidence indicates the production was deliberately stuffed with duplicate files to impede discovery.

D6. The Development Production was clearly fabricated using test files irrelevant to support the phony Teytel Algorithm

107. Dr. Polish's Second Expert Report at 6, 14 (Ex. NN), stated that REDACTED REDACTED REDACTED Examining this key codREDACTEDwe find the following line of code

REDACTED

108. ThiREDACTEDis clearly a code for LDS100 paths, not a code for LDS200 200 paths. This is the only script codREDACTEDn the entire Development Production. Dr. Teytel could not have gotten off on his first step of LDS200 development with this script code. Dr. Teytel messed up in his fabrication. Placing LDS100 development code in their key 200 path sequence development code shows clearly the files are made up. Defendants completed their development and installation of LDS200 in 2000. In his Second Expert Report (Ex. NN, p.11), Dr. Polish provided a table with columns showing REDACTED the REDACTED REDACTED and REDACTED . In the REDACTED REDACTED REDACTED I tried to find the REDACTED . However the Production containsREDACTEDfiles ofREDACTED (in the folderREDACTED). Each file is REDACTED For seedsREDACTEDeach file contains REDACTED REDACTED However these REDACTED were clearly not used for selecting seeds but were apparently used for REDACTED REDACTED There are also REDACTED REDACTED purportedly tested by Dr. Teytel and confirmed by Dr. Polish. I found only REDACTED REDACTED Polish claimed that he verified that REDACTED seeds have been calculated. Dr. Polish's report used the REDACTED as the example to show how the seeds were selected. However, there are no REDACTED for even this exemplary REDACTED REDACTED Of the REDACTEI eeds selected by Teytel in the REDACTED we only find REDACTED

having REDACTED All othe REDACTED REDACTED; appear to have been chosen out of blue. It could not be that because of the lack of space that Dr. Teytel did not save the REDACTED for the selectec REDACTED Dr. Teytel even saved mostly meaning processing message outputs when the REDACTED According to Dr. Polish's Declaration for each seec REDACTED

109. In folders REDACTED REDACT we find more records of REDACTED REDACTED For eack REDACTED presumably for a range of seeds. There were REDACTED REDACTED It appears that Defendants did not care what REDACTED . This, again, is consistent with the idea that the calculations were done to compute some benchmarks of a CMO portfolio to test some predetermined sequences (like ACE derivative sequences) and not to select sequences. The averages of prices of CMOs alone, without the seeds used to produce the prices, would be of interest only for benchmark purposes.

110. These purported unverifiable data, if they were calculation of prices, could have been done any time before July 2005. This was one more reason for Dr. Teytel to modify all the "Modified dates" of these files when he packed these benchmark calculation price outputs and masqueraded them as seed selection based on CMO prices.

111. In addition, Defendants could not have tested REDACTED as they claimed. Dr. Teytel testified that it takes approximately REDACTED REDACTED, REDACTED REDACTED REDACTED to complete the work, much longer than REDACTED claimed by Defendants to develop their LDS200 sequences. This estimate does not include the REDACTED REDACTED outside Teytel's seed range of REDACTED REDACTED REDACTED to run. All of these REDACTED of calculations were done for no reason related to Dr. Teytel's seed selection efforts.

112. The Polish Second Report repeatedly observed, see pp.9-10, Ex. NN, that REDACTED Now we just saw that Dr. Teytel ran ove REDACTED REDACTED to calculate the benchmarks of bonds. Dr. Polish's statement above confirmed that Teytel did not follow his Mixed Seed Algorithm for LDS200 development. Otherwise, he would not have run all these benchmarks that were unnecessary to his purported Agorithm.

113. I found in the folder REDACTED There are no CMO prices, so the scores were out of nowhere and cannot be verified. Dr. Polish's efforts to lend credibility to this entirely deficient Development Production would not be found acceptable in the mathematical community. It is almost as bad as the outright false statement in his first Declaration dated February 28, 2006, ¶ 5 that "...after reviewing the RCS files produced by Defendants, I was able

to determine, for any given day during the relevant period, all of the Yield Book Numbers generated during the relevant period. Mr. Hicks should be able to do the same." (Ex. QQ) In response, Mr. Hicks pointed out in his second declaration dated June. 8. 2007. ¶¶ 16 that the code contains an REDACTED REDACTED REDACTED that Dr. Polish had admitted in a telephone conversation that he did not know what inputs into REDACTED were used and that he was simply trying to show that numbers could be generated by the Yield Book code in general. Ex. ZZ.

114. The first Polish declaration was at best misleading. The justification he raised, that he was simply trying to show that numbers could be generated by The Yield Book, was irrelevant. The issue then and now is not whether Defendants' code can generate numbers, it is whether the numbers generated are the numbers that were in fact used in production. His first Declaration tried to give the impression that he could determine the sequence used in production from the files produced. In an obvious face saving effort, Dr. Polish responded with Second Declaration and in ¶ 10 cited a May 22, 2007, letter from defense counsel, unsupported by a sworn statement from anyone with first-hand knowledge, to the effect that REDACTED REDACTED Ex. YY. Further, Dr. Polish admitted he has no idea whether the Teytel Algorithm is a good way to pick seeds or price securities or not. Polish 7/16/12 Dep. 46:6-47:11, Ex. AAA. The above analysis shows that the Second Polish Expert Report provides no support for the proposition that the Development Production is genuine.

115. Based on all the evidence outlined in Dr. Polish's Second Expert Report pertaining to the Development Production, the outputs were likely to all have been created or pulled from irrelevant files together in 2005 to masquerade as purported independent development efforts using Teytel's algorithm. That would explain Defendants' 6 year fight to resist their production. Of the first REDACTED of the Teytel's Notebook were abou REDACTED REDACTED which stopped when Dr. Wang confronted Defendants about their breach in January 1999. The purported development of LDS200 described in Teytel's Notebook (Ex. SS) which purportedly was evidenced by the documents in 4 of 5 DVD's produced (CGM07077-80) occupied REDACTED is very sketchy, does not described much or say much, lacks any details allowing verification, anc REDACTED REDACTED which itself has large deficiencies. I found the unverifiable Production just added one more suspect strong reason to conclude that the Sequence Development File and Testing File Production was fabricated to cover up the use of ACE. The unverifiable Production certainly does not support the unverifiable sketchy Teytel Notebook entries for LDS200 in any way.

116. The Production contained not a single set of test code files to allow anyone to compile, link and build a The Yield Book application, any application to test any sequence. Defendants still maintain the RCS code, which is the archive of The Yield Book code. From a complete set of RCS code, one can extract the testing files purportedly used by Defendants. So a complete set of RCS code contains the missing testing files used by Defendants and could easily have been produced. That fact that they resisted indicates that this evidence would have been harmful to their case.

D7. Defendants destroyed all of Radak 1000 Path Sequence development files and testing files that would confirm that Russell 1000 Path Sequence does not exist. Defendants also withheld all ACE systems that would reveal their theft of ACE.

117. As we discussed above in Section C, during the depositions on August 29, 2012, Radak testified that he deleted all of the development files and test files for his 1000 path sequence, in April 2006 during this litigation, Ex. FF. It is against Defendant's own file saving policy. This destruction appears to be consistent with Defendants' pattern of systematic destruction of incriminating evidence, regarding other sequences, including LDS64, LDS100 and LDS200.

118. In addition to the offline ACE Use Code, Defendants withheld the ACE test system files. Defendants claim that "After Defendants and AAI failed to reach agreement on a license for the ACE Numbers, Defendants began working internally to develop a means to reduce the number of paths used in the Monte Carlo simulation component of the Yield Book valuation model, without compromising the model's accuracy. Defendants' 56.1 Stat. of 2008 ¶ 77. Teytel was assigned to work on this project. Id. After several months of work, Dr. Teytel developed an algorithm". See Defendants' Memorandum of Law of Defendants in Support of Their Summary Judgment section E., Defendants' Development And Implementation Of The Yield Book Mixed-Seed System, at 8. The purportedly Teytel Mixed Seed Algorithm was developed before November 3, 1998, see CGM178. Since Dr. Teytel was not "assigned to work on this project" until after "Defendants and AAI failed to reach agreement on a license", that means that Defendants were no longer interested in an ACE license in July 1998 which was, as Defendants stated in the above excerpt, "several months" before Teytel developed the Teytel Algorithm. Consistent with this, Hayre also testified that when Dr. Wang called him to arrange the February 1999 meeting, Defendants had decided that they would not license ACE and did not want to want to "hear another sales pitch". Hayre Dep. 257:7 to 258:12, 259:24-25.

119. Mr. Russell testified that he deleted all ACE testing files "within a month or two" after Defendants concluded that they were not going to deal with Dr. Wang. Russell Dep. 150:24-151:22. As shown above, that was by July 1998, several months before November 3, 1998, when the Teytel Algorithm had already been developed. So Russell's testimony amounts to a claim that he deleted all the ACE testing files by approximately August 1998. Yet Dr. Teytel testified that Mr. Russell gave him access to the same ACE test system[25] and that he used it eight months later, in or after April 1999,[26] to do an "ACE64 comparison for LDS100". Teytel May 17, 2007 Dep. 183:18-184:3, 231:16-234.6. Dr. Hayre testified that Mr. Russell and Dr. Teytel together reported the LDS100 test results to him. Hayre Dep. 103:13-25. Both Russell and Hayre were advised of the results. See Russell's August 26, 1999 email, AAI 040. Because that decision was in approximately July 1998, and Defendants were using the ACE test files in April 1999 to test against ACE, clearly Russell provided false testimony at his deposition in 2007, when he claimed that the ACE test files had been deleted "within a month or two" of when Defendants decided they were not interested in licensing ACE. Further, Defendants Answer to

[25] Teytel Dep. 230:13-234:6
[26] See defendants' Answers and Objections dated January 25, 2006 to plaintiff's Second Set of Interrogatories etc., Responses 1-4.

Complaint and Counterclaim, ¶ 82, stated "Dr. Wang contacted Dr. Hayre in or about February 1999 and accused defendants of misappropriating ACE". Ex. YYYY. In September 1999, AAI's counsel wrote to Defendants accusing them of reverse engineering ACE. In February 2002, AAI's counsel sent a draft complaint to Defendants and the suit was filed in 2004. (Def. Rule 56.1 Statement ¶¶101-104, Ex. PP) Clearly, Defendants were aware of AAI's claims continuously since at least February 1999. In light of this, any deletion of records relating to ACE, such as ACE test systems and Sequence Selection Code could only have been for the purpose of covering up wrongdoing.

120. Russell also testified that he did not save term structures during the 4th ACE test[27], but this was exactly what he did using the ACE Theft Code he designed, as demonstrated in Section E8 below.

E. <u>The ACE Theft Code in The Yield Book</u>

The RCS archive files produced by Defendants show that Defendants installed the first version of code designed to steal ACE ("ACE Theft Code" or "Theft Code") *in The Yield Book code files on August 1, 1996. They then called Dr. Wang the next day to invite him to test his ACE sequences using The Yield Book. The files produced show that during a period of one and half years, Defendants revised the ACE Theft Code many times in lock step with their progress of testing ACE. Then finally, prior to the 4th test of ACE on June 3, 1998, Defendants compiled ACE Theft Code into The Yield Book system which had been modified to test the ACE sequences ("ACE Test System"). During testing, the ACE Theft Code surreptitiously saved and stored in a hidden directory all the interest rate paths generated by the ACE sequences. The Theft Code was also designed to allow Defendants to retrieve the ACE Interest Rate Paths to back out all ACE sequences.*

121. I have identified the code in the RCS code that was designed to steal ACE ("ACE Theft Code" or "Theft Code"). The Theft Code is tucked away in totally unexpected locations among Defendants' approximately 3,400 RCS code files containing hundreds of thousands of lines of code. Flow charts show how The Yield Book used sequences to generate sequences before ACE, during ACE testing when the Theft Code was used, and after the theft had occurred are shown in Figs. 1-3 of Ex. SSS hereto. As shown in Fig.1 of Ex. SSS, before ACE testing, a sequence produced by the sequence generator is fed into the Interest Rate Model part of The Yield Book system which, in turn, generated a set of interest rate paths. These interest rate paths are then fed into the Cash Flow Generator. The Generator generates cash flow for each interest rate path for a particular security or deal. The interest rate paths are then used again after adding an OAS to discount the cash flow from each path to get their present value, and the present value of the cash flow for all the paths are averaged together to get a value Robert Russell Decl. dated Feb. 12, 2008, ¶3, Ex. BBBBB. As you can see in Fig.1, the interest rate paths are used twice. Therefore, errors in the interest rate paths can have a large effect on the accuracy of the calculation of the price.

[27] See Defendants' motion for summary judgment of 2013 at 4, citing their 56.1 Statement para.20, which cited Robert Russell Decl. dated Feb. 12, 2008, Ex. BBBBB.

122. Fig.2 of Ex. SSS shows the ACE Test System as rigged with the ACE Theft Code. The functions in brown in the flow chart of functions calls in Ex. CCC, illustrated part of the functions in the ACE Theft Code added on August 1, 1996. Dr. Wang supplied the ACE sequences themselves for testing. The sequences were supplied in data files on floppy disk that were inserted in the floppy drive of the standalone computer on which the test was run for "security" purposes. Robert Russell Decl. dated Feb. 12, 2008, ¶¶4-5, Ex. BBBBB. Once the calculations were completed and the prices determined, the ACE Interest Rate Paths were supposed to be discarded so that the ACE sequences could not be backed out of them. The use of the ACE sequences was controlled by the use of a control flag called "-xiaolu", Dr. Wang's first name28. The –xiaolu flag tells the program to switch off the sequence generator and switch on the process of reading sequences on the floppy disk, the ACE sequence. 3rd Wang Decl. ¶11 Ex. BBB. This is shown in Fig.2. by the figures enclosed in a solid line on the left hand side.

123. The Theft Code is shown in the right side of Fig.2 by broken lines. As explained below, the ACE Theft Code was designed to save the ACE generated interest rate paths, permit them later to be retrieved and used with the Interest Rate Model to back out the sequences. The Theft Code was tucked away far down in the computing chain of action long past the step of feeding the sequence into the Interest Rate Model in another part of The Yield Book that was designed to pass the interest rate paths to the cash flow generators. The first appearance of The Theft Code occurred on August 1, 1996, the day before Defendants called Dr. Wang on August 2, 1996, to ask him to test his sequences in The Yield Book. 3rd Wang Decl. ¶ 6-7 (Ex. BBB). It is found in the code file REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED is illustrates in Figs.1-3, Ex. SSS. In a version, 1.21 of REDACTED, revised April 8, 1998 (less than two months before the last test of ACE when it was stolen)(Ex. DDD), the Theft Code occupied most of that file - the first four of seven pages. The goal of the Theft Code was spelled out in the REDACTED on the first page: "save_seq", i.e., to save the sequences run by The Yield Book. The sole purpose of "save_seq" is to "save" the sequence, the ACE sequence being tested, by saving the ACE Interest Rate Paths generated by the ACE sequence during the testing so that later. the ACE Interest Rate Paths could be retrieved by using another Theft Code function " REDACTED literally REDACTED the ACE Interest Rate Paths, and backing out the ACE sequences as discussed below.

124. That the ultimate goal of the Theft Code is to save sequences, rather than save interest rate paths, is supported by the terminology used. There are only REDACTED major functions defined in the Theft Code, REDACTED and REDACTED [29], that contain the words REDACTED .

[28] The use of the hyphen, "-", before the code name indicates it is the name of an option.

[29] A "function" is a predefined set of source code instructions that tells the computer to carry out a particular action, e.g., save a set of interest rate paths that had been generated in RAM or get a set of interest rate paths that had been saved to a file. A function can include other functions.

But there are many references to the function called "save_seq". In one section alone of Theft Code, there are REDACTED references to the function "save_seq" within 22 lines of code in REDACTED v1.21p.230, Ex. DDD. "save_seq" is not just the ultimate goal of the entire Theft Code, it is also the REDACTED v1.21. This demonstrates that Defendants were obsessed with saving sequences, and not just any sequences. Hence the import of Robert Russell's login REDACTED

was often used by Defendants, and particularly Robert Russell who wrote most of the ACE Theft Code, to REDACTED clearly refers to Dr. Xiaolu Wang, Plaintiff's principal and is a natural extension of the "-xiaolu" name for the control flag he and Mr. Russell used in the ACE Test System code to read in the ACE sequences and switch off Defendants sequence generator. Further, as Defendants admitted33, there was no reason for them to save sequences generated by their sequence generator, Fig.1, Ex. SSS. That is because they could regenerate it at any time with their sequence generator much faster than they could read it from a file.

For example, a function to save interest rate paths, could contain other functions to take the interest rate paths from RAM and save them to a file in a directory specified by the programmer when this function is run.

[30] REDACTED REDACTED

[31] The RCS archive files contain all revisions of a given file that is properly logged in using the RCS program. The RCS file is denominated by have a REDACTED at the end. So the RCS version The Yield Book code file REDACTED When a revision to a file is properly saved, it is logged in to the RCS file. At the time, the programmer has the opportunity make a comment about the revision that is logged in. Hence the term login comment. Login comments are purely optional.

[32] Robert Russell explained that the Term Structure Model portion of The Yield Book generates interest rate paths. Russell Sept. 18, 2007 dep. 300:15;18, REDACTED

And Mr. Russell referred to interest REDACTED rate paths generated by the term structure model as the "term structure". Dep. 91:3-92:5, REDACTED

REDACTED

(Emphasis added.)

[33] March 1, 2006 Hrg. Tr. 12:20-21 "The random numbers [this is the term Defendants use most often to refer to sequences] themselves are not saved on a daily basis by defendants. There is no reason to do that...."

125. The evolution of the Theft Code is shown in various revisions of the Mortgage Research RCS code ("MR-RCS"). Each revision is numbered and indicates the date and person who created it, which was always Robert Russell except right after ACE theft in June 1998 when Stuart Herman on one occasion helped to install code to use the stolen ACE for The Yield Book production.

E1. The Theft Code that Defendants used to steal ACE

126. In the ACE test systems, the flag (a flag is a signal in the code for the computer to do something) for compiling34 the code to read in the ACE sequences was named "-xiaolu". (Wang Decl. ¶_11, Ex. BBB) When running with the "-xiaolu" option set to be TRUE, i.e., nonzero, the ACE test system would read in the ACE sequences. The ACE Theft Code was launched under the flag simply named "-x", followed by the characters: "seq" to indicate that sequences were the ultimate object to be stolen. The path to locate the file containing stolen ACE interest rates35 is specified by the environmental variable "path_dir" and must be input either through a key board or in a script file36 each time the program is started. The script file is not disclosed in the code produced. This permits Defendants to hide the location of the stolen interest rates and prevents Plaintiff from examining them. After the ACE sequences were stolen, the ACE Use Code was installed right away and the following very telling log comment was added to pathgen_cover.c version 1.24: "@hide PathID from the Yield Book" (emphasis added). See pathgen_cover.c,v p.19, Ex. EEE. This is discussed further below.

127. The RCS code shows that while Dr. Wang was conducting the ACE tests with Mr. Russell, from Defendants' first call by Dr. YK Chan to Dr. Wang on August 2, 1996 to ask him to test the ACE sequences (Ex. BBB ¶ 7) until the last ACE test, on June 3, 1998 when Defendants stole ACE, Mr. Russell was fine-tuning the Theft Code in 14 revisions over a period of almost two years.

128. Defendants took advantage of Dr. Wang's own code designed for the ACE test system to steal and later use ACE. Dr. Wang coded the ACE test code to use a "flag" to read in the ACE sequences from the ACE data files and then to use them to generate the ACE Interest Rate Paths. (Ex. BBB ¶ 11) In a nutshell, as will be shown in detail below, the flag "-xiaolu" in the ACE test system allows the substitution of the ACE sequences stored in a data file for the sequence generated by The Yield Book sequence generator to feed into The Yield Book interest rate model ("Interest Rate Model" or "Term Structure Model") to generate ACE Interest Rate Paths. The Theft Code was a modification to the ACE test system. After the –xiaolu flag in the test system was used to read in ACE, a REDACTED in the Theft Code surreptitiously saved the ACE Interest Rate Paths during the last ACE test. Next the Theft Code permitted Defendants to REDACTED

[34] Computer code is first written by programmers in an editable text file which is called "source code". The source code must be turned into code that the computer can read and execute. This is done by using standard compiler software that reads the source code file and "compiles" it into computer readable code called "object code.", and then links the collection of object code into either an executable code, or a library.
[35] An example of a hypothetical "path" to tell a program where to find a particular "file A" might look like c:\interest rate files\file A.
[36] A script code file is a source code file that the computer can read and execute without compiling.

the ACE Interest Rate Paths and use them with the Interest Rate Model and its inputs to back out the ACE sequences. For both the Theft Code and the Use Code, Defendants used a REDACTED to trigger the desired action. The REDACTED uses hidden REDACTED to store the directory path[37] containing the saved ACE Interest Rate Paths. There was no need for Defendants to use REDACTED in connection with this action except to conceal what they were doing.

129. In essence, Defendants took Dr. Wang's ACE test system code (a1) of Table A (in the Top Level Description), which was designed to read-in the ACE sequences and to generate ACE Interest Rate Paths, and surreptitiously modified it and used it in conjunction with ACE Theft Code (a2) and (a3) to steal ACE, and then later modified the function (a1) to function (b1) of Table B to use stolen ACE. See Section F below for an explanation of the ACE Use Code.

130. The ACE Theft Code hidden from Dr. Wang (a2) and (a3) of Table A, is found in an inconspicuous code file called REDACTED It comprises two parts, a part (a2) to save the ACE Interest Rate Paths and a part (a3) to retrieve them. On August 1, 1996, REDACTED REDACTED by Robert Russell.[38] REDACTED The login comment says: REDACTED

131. The REDACTED REDACTED to store ACE rate paths. REDACTED REDACTED Then the function REDACTED ACE rate paths. This latter function is found from REDACTED REDACTED REDACTED On June 5, 1997 there was a REDACTED by Mr. Russell, with log comments.

132. The REDACTED REDACTED where ACE rate paths have been stored and to REDACTEDREDACTED them. This code is found from REDACTED REDACTED (Ex. EEE).

133. In each of the functions, the REDACTED of ACE rate paths is REDACTED declared by the code:

REDACTED

(page 19) where REDACTED of the ACE sequence. All the Theft Code in The Yield Book was under the preprocessor REDACTED which allow access only by the **inner circle** people who run the ACE test, backed out ACE sequences and use the ACE

[37] The "REDACTED indicates the name of a file what directory it is located in so the computer can find it when executing a program.
[38] In fact, the programmer ID for REDACTED v 1.7 was "russell" which was clearly a reference to Robert Russell as he was working on the code at that time and the record indicates no one else with the name "Russell" performing that function at the time.

sequences to generate interest rate paths **offline**. Since Robert Russell wrote the Theft Code and supervised the testing of ACE, he appears to be one of the **inner circle** people. The preprocessor REDACTED was concealed from Dr. Wang during the last ACE test.

134. Defendants used the application, REDACTED, during the last ACE test to save ACE term structures in the 4th ACE test. See E1 and E8. The Theft Code (Ex. DDD, REDACTED v1.21) operated as follows:

(1) Robert Russell supervised ACE testing. (Russell Decl. Ex. ¶4, BBBBB,) The login comments in the RCS files indicated that Mr. Russell was responsible for most of the relevant changes to the code. He launched the rigged ACE test system with REDACTED option, for a simple example, say REDACTED to save ACE64 rate paths for just one scenario. He also set value of REDACTED to be the hidden path directory to store ACE rate paths. The ACE test system read in an ACE sequence from ACE data files provided by Dr. Wang, and used it to generate several sets of ACE rate paths.

REDACTED (2) REDACTED; was run with REDACTED option turned on, which told the program to ACE term structures[39]; then in REDACTED

REDACTED

REDACTED

(3) Now ACE rate paths were already in the RAM[40] of the computer, after the function REDACTED

REDACTED

REDACTED

[39] The code file REDACTED or other The Yield Book application, REDACTED During the 4th test, cmoopt was also installed ACE Theft Code. Defendants could have used cmoopt instead of REDACTED to price IO's and PO's to steal ACE, see E8 below.

[40] RAM stands for random access memory and is the place where all the computer's calculations take place.

REDACTED So the ACE rate paths are "saved". This code is encapsulated beginning with the condition REDACTED REDACTED which indicates that the code is off limit to The Yield Book users and is for the people in the **inner circle** who could access the stolen ACE.

(4) After ACE tests had been completed, the rigged ACE test system was launched again with REDACTED turned on to RED the REDACTED ACE term structures. It found the hidden path directories that stored ACE term structures, and then called the function REDACTED

REDACTED

(5) Then each of the tested ACE sequences was backed out from the saved multiple ACE rate path files. We shall discuss this in detail below.

135. The Yield Book generates each interest rate path from a path of the sequence in the following manner. REDACTED

136. This formula shows that at REDACTED . Conversely, once Defendants stole sufficient ACE rate paths, REDACTED , they could reverse all REDACTEI and then from the equations above, find REDACTED which are the pair of ACE numbers at the time step for that path. After this has been done for all the saved rate paths generated from one ACE sequence,

41 REDACTED See Teytel August 23, 2013, dep. Q. REDACTED

that entire ACE sequence has been backed out.

137. The last ACE test on June 3, 1998 ran through at least 6 different scenarios of yield curves at Defendants' demand (Russell to Dr. Wang email: **REDACTED** CGM00059, Ex. GGG. For every scenario, Defendants used Theft Code and **REDACTED** if both swap curves and treasury yield curves were used. In total there are at least **REDACTED** for each ACE sequence tested. These many sets of ACE term structures saved by Theft Code were more than enough to back out the entire ACE sequence, in a clean sweep.

E2. Defendants prepared ACE Theft Code to steal ACE sequences, and then the next day invited Dr. Wang to do tests

138. Dr. Lakhbir Hayre, who was then the Managing Director and the Head of Defendants' Mortgage Research, expressed intense interest in Dr. Wang's pioneer work in Deterministic Simulation Blaster™, a Monte Carlo simulation technology Dr. Wang helped to develop for IBM, Wang Decl. ¶ 5, Ex. BBB. Dr. Wang declined Dr. Hayre's invitation and informed him that he has resigned from the position of senior scientist of IBM in early February 1998 to work for Plaintiff focusing on new research on sequences (Wang Decl. ¶ 5, Ex. BBB). Through Dr. Y.K. Chan, Defendants' in house leading expert in term structure modeling and Monte Carlo simulation and his friends, they closely monitored Dr. Wang's progress.. When Dr. Chan heard that Dr. Wang made a new breakthrough in July 1996 (Wang Decl. ¶ 6, Ex. BBB), Defendants promptly directed their master programmer, Mr. Robert Russell, to design the Theft Code hidden from Dr. Wang, to "save-and-get" the sequences that would be tested by The Yield Book in Defendants' standalone machine.

139. The Theft Code (e.g., in `pathgen_cover.c` and `mainshell.c`, Ex. s EEE and FFF) was first installed by Robert Russell, on August 1, 1996, one day before Defendants called Dr. Wang to arrange ACE tests. At that point in time, Dr. Wang had not even finished testing ACE nor had he given it a name. (Wang Decl. ¶ 7, Ex. BBB).

E3. Defendants ACE Theft Code clearly specified that its goal was "save_seq" and that the sequences they intended to "save and reuse" was not their one and only sequence, the Production 200 in use when they were testing ACE, it was ACE.

140. ACE Theft Code was run under the **REDACTED**. Defendants' reminder for REDA clearly indicated that not their Production200, their one and only "seq" (sequence) at that

[42] As shown by the debug message on **REDACTED**

time, but that Plaintiff's ACE "seq"s were the object that attracted them, their goal to "save and reuse". Both of Defendants' code files, REDACTED which REDACTED and the code file REDACTED See the color flow chart, Ex. CCC), had the REDACTED installed prior to the last test of ACE in June 1998. In REDACTED less than 2 months before the last test of ACE. In REDACTED (See Ex. FFF, p.32). The REDACTED

This is the version (See Ex.DDD, p.1) used for the 4th ACE test of June 3, 1998, during which REDACTED was used to surreptitiously "save" ACE term structures when it was pricing passthroughs (E8).

141. Here REDACTED RED the flag name for the Theft Code. The REDACTED indicates that the goal of the Theft Code is REDACTED the sequence - indicating the target and sole purpose of REDACTED is to get the sequence. The code then confirms what Defendants admitted in court that there is no reason to save Defendants' own one and only sequence, the Production200, see E5 below. So, the Theft Code only kicks in to "save", i.e., steal, REDACTED in other words, REDACTED The circumstances indicate that REDACTED ACE.

142. Defendants had only one sequence: the single seed 200 paths Production200, from October 1, 1994 through April 1999, other than the ACE sequences which were tested and "saved" by Defendants June 3, 1998. Defendants Answers and Objections to Plaintiff's Second Set of Interrogatories, Interrogatory Answers 1-4, Ex. W, Radak Decl. ¶¶ 2-4, Ex. O. Clearly, Defendants did not go to great trouble to painstakingly design the Theft Code to steal their own sequence. They sought to "save" the rate paths generated by very specific "AABB sequence". If Defendants sought to save their one and only sequence of 200 paths, the "if" part in "if seq is AABB" would not have been included and makes no sense. It certainly makes no sense to begin installing this code to save their production sequence on August 1, 1996, almost two years after they first used it in production and one day before inviting Dr. Wang to test his sequences at Citigroup and then revise the code in June 1997 while they were in the process of testing ACE. It only makes sense if there is another sequence to be saved, another sequence that they wanted to save. The only other sequences in sight were Plaintiff Advanced Analytics (AA) sequences which Dr. Wang later named the ACE sequences. So the "AABB" sequences that Defendants have designed the Theft Code to "save and reuse" could have been no other sequences than the ACE sequences that they were planning to test[43] and for the reasons discussed below and in Section F.

[43] Russell Dep. 9/17/07 198:10-14 - Q. Were you testing any other sequences during the period you were testing Dr. Wang's sequence? A. I don't recall other testing -- testing with other sequences.

143. When on August 1, 1996, Russell was installing the Theft Code, Dr. Xiaolu Wang had not yet created the name ACE for his new sequences. The name ACE didn't exist. Through Dr. Chan, Defendants only knew the name of the Dr. Wang's corporation was Advanced Analytics (AA). Defendants also knew that Dr. Wang had been the principal inventor of IBM's (Deterministic Simulation) Blaster. Defendants averred in their defense that they believed REDACTED sequences (ACE) were, or were somehow derived from IBM's Blaster sequences, i.e., REDACTED sequences. So it is logical that REDACTED ' became Defendants' code name for AA's sequences before AA coined the name ACE.

<u>E4.</u> There was no reason for Defendants to "save and reuse" their own sequence, only saving ACE sequences makes sense.

144. There is no reason for Defendants to save their own sequence to a file[44] and then reuse it because it is much faster and more robust to save and use their sequence in RAM with their Monte Carlo code. There is even less reason to save and reused the term structures (interest rate paths) that could be generated by their own sequence to a file because it is tens of thousands of times faster and more robust to generate their sequence in RAM with their Monte Carlo code and use it to generate interest rate paths, than to read the term structures from a file stored on a hard disk. The fact that Defendants were reading in term structures from a stored file on a daily basis (see Section F) leads to the inevitable conclusion that they were doing it to surreptitiously switch their own interest rate paths for ACE Interest Rate Paths, and that the code withheld by Defendants uses the stolen ACE to generate ACE Interest Rate Paths offline.

145. Right after the ACE theft, Defendants default method of getting interest rate paths was changed to read them in from a data file (See Section F). This makes no sense if Defendants were using the interest rate paths generated by sequences generated by their own code. Writing to and then reading in the same interest rate data from a large data file would accomplish nothing beyond making the process thousands of times slower and more error-prone than generating the interest rate paths into computer RAM. But if Defendants are using the ACE sequences, this method is essential because Defendants cannot generate the ACE sequences. Defendants stole only ACE sequences in data files. They can only use the ACE sequences verbatim by reading them in from a data file and then using them to generate interest rate paths. So to use the ACE sequences verbatim in their Yield Book, it had to either read in the ACE sequences or read in interest rate paths generated by the ACE sequences. Defendants did the latter which was more burdensome for them but it provided more stealth.

146. Further, the analysis of ACE Use Code below will indicate that reading in the ACE generated interest rate paths was the primary method used by Defendants' traders in production. This makes perfect sense. The use of the ACE sequences verbatim would be preferable for their traders to the use instead of Defendants' ACE derivative sequences that they obtained through targeting the ACE sequences, which would be expected to be less accurate.

[44] March 1, 2006 Hrg. Tr. 12:20-21 "The random numbers [this is the term Defendants use most often to refer to sequences] themselves are not saved on a daily basis by defendants. There is no reason to do that…."

147. Defendants also admitted that there was no reason to save Defendants' one and only sequence (which Defendants call "random numbers"[45]) used from **REDACTED** . In the March 1, 2006 court hearing, Defendants averred "The random numbers themselves are not saved on a daily basis by defendant. <u>There is no reason to do that.</u>"[46] So they were not trying to save their own sequence, they were trying to save ACE.

E5. Defendants have confirmed that other than the ACE sequences and their own sequences, they never had access to any other sequences

148. Defendants have confirmed that they have never tested anyone else' sequences other than AA (Advanced Analytics)-sequences: ACE. Russell Dep. 9/17/07 198:10-14 REDACTED

”

E6. ACE Theft Code was customized to capture the entire length of an ACE sequence.

149. It appears that Defendants had prepared the first version of the Theft Code of August 1, 1996 hastily, before calling Dr. Wang the next day after its installation. It even had a typo within the code which was unusual for Defendants' RCS code, mistyped "`dyc`" (the current market yields) for "`paryc`" (the term structures generated by the sequence from perturbing dyc. see Line 330, `pathgen_cover.c,v` , Ex. EEE), made it unworkable. This was consistent with the fact that the Theft Code had not been "battle"-tested by a sequence. See v 1.7 Ex. JJJ. Russell fixed the bug 5 days later in v 1.8 (Ex. III), (log in comments: "`@correct typo`").

150. Since v 1.7 of August 1, 1996, the code file pathgen_cover.c,v (Ex. EEE) has been revised 14 times until 1998/4/8, v 1.21. (Ex. DDD). All of its 14 revisions but one (v 1.13, a very small change) were done exclusively on the Theft Code. All of the 13 revisions to the Theft Code but three (by Dr. Hans Schmitt – v 1.17 which made little change, v 1.20 and v 1.21) were drafted by Mr. Russell who dealt with Dr. Wang during the testing of ACE. Most of the revisions appear to be made to facilitate the theft of ACE sequences (See Section E7 below). It appears that Defendants were not dealing with Dr. Wang in good faith from the very beginning.

151. As discussed earlier in section B above, the dimension, i.e. the number of rows, of all the sequences generated by TYB Monte Carlo code is always Redacted However, the first technical issue encountered by the designers of Theft Code, was that for the sake of efficiency and depending on the maturities of the batch of the specific deals sought to be priced, The Yield Book system does not always load `paryc` with *term structures* generated using all of the Redacted dimensions of a sequence.

[45]This was what Defendants misleadingly called the Sequences they used in the court "Yield Book numbers". Yet refused to confirm either the definition or that the sequences they produced were genuine in Radak Declaration as discussed Section B above.
[46] Tr. 12:20-21

152. In order to capture all REDACTE dimensions of the ACE sequences, Defendants used deals with "*late*" maturities, such as 40 years, or manually change the deal maturities dates to 40 years, and make sure 30 years Treasury bonds, not only the usual 10 year treasury benchmark prices for MBS, are used. Then The Yield Book would have to use all REDACTE rows of ACE sequences to generate the term structures. This consideration explained the initial installation of Theft Code in the unlikely code file REDACTED REDACTED Defendants first added:

REDACTED

to V1.39, p.1, and then added functions such as the function call REDACTED REDACTED in ACE Theft Code, to make sure that term structures out for 30 years of future curves will be generated so they could capture the entire ACE sequences[47].

153. The function REDACTED is installed on REDACTED REDACTED by Mr. Russell. REDACTED REDACTED

REDACTED

154. The other difference between v1.39 on 1996.11.13. and v1.38 of 1996.10.4 of the code file REDACTED are follows. REDACTED

REDACTED

REDACTED

4

REDACTED
REDACTED

[48] A function is called a `subroutine` in Fortran, "Formula Translation", programming language:

REDACTED

REDACTED

REDACTED

This means that 30 year treasuries were added. Also there is an insertion in V1.39, REDACTED
REDACTED

155. Again on 1996.11.13, Russell also installed v.23 of REDACTED . Its difference with v1.22 of REDACTED installed by Y.K. Chan REDACTED

REDACTED

156. There REDACTED REDACTED
The only function that Defendants REDACTED REDACTED to
remind them why they were REDACTED meaning
use the saved ACE Interest Rate Paths.

E7. The Theft Code was developed in lock steps with the progression of ACE tests

157. The Theft Code revisions appeared to progress in lock step with the progress of the ACE tests until after the last round of ACE tests on June 3rd, 1998. Right after that, the ACE Theft Code was upgraded to ACE Use Code to cope the more rigorous production demand in The Yield Book. The ACE Theft Code had to be used only once. In production, the ACE Use Code would have to be used daily or even more often for calibration. From the emails it appears that for each ACE test, Dr. Wang would run ACE sequences in a single day and then Russell would run large batches of several thousand seeds of Monte Carlo simulation that would take weeks. Then Russell would average the thousands of Monte Carlo results, e.g., REDACTED ,

REDACTED ;, and compare them with the ACE results. Had Russell disclosed the ACE Theft Code to Dr. Wang during the ACE tests period, Dr. Wang clearly would not have been willing to proceed with ACE tests.

158. On 3/21/97, Russell emailed Dr. Wang and encourages Wang to bring a "disk with some code and data to check out whether we can read a disk that you produce. Of course, such a disk should not contain any proprietary information at this point in time." Ex. MMM. This is consistent with the fact that at that point in time, Theft Code in REDACTED at v 1.13 was not yet "production"-ready. It was not stable enough and it still needed to save more term structures (interest rate paths) in order to back out ACE sequences. It appears that Russell had been busy setting up the 1st ACE test at about that time.

159. The first ACE test using ACE32 appears to have been successfully completed by the end of April 1997. Then a significant upgrade happened in Theft Code. From v 1.14 of 1997-4-28 to v 1.15 of 1997-6-5 of REDACTED The log comments to v 1.15 was REDACTED Now the Theft Code could REDACTED ' the term structures generated by multiple ACE sequences in multiple REDACTED (These directory "paths" are unrelated to rate "paths"), specified by hidden REDACTED which was virtually undetectable by Dr. Wang[49].

160. The Theft Code in RE at v 1.15 (Ex. NNN) also may "save" multiple term structures generated by one sequence from just one yield curve. This cut down significantly the number of "scenarios", namely, yield curves, that The Yield Book must run to back out ACE. Following the code line

REDACTED REDACTED

v 1.15 added a block of code to allocate the memory for directory names of the multiplied files of "saved" term structures[50].

[49] RED v 1.14 (Ex. XXX): REDACTED
V 1.15 (Ex. NNN) REDACTED

REDACTED REDACTED

the Theft Code added the lower half page code on p.2 of Ex. NNNN, V.1.15. REDACTED
REDACTED

REDACTED

161. *v 1.15* of 1997-6-5 has bugs. A correction was made on V1.16 on 1997-6-9 (log in comments:" REDACTED ")(Ex. s NNN and <>). It was likely deemed production ready after more tests. Then Russell emailed Dr. Wang on 1997-6-12 (CGM033, Ex. OOO) to urge him to come in to test ACE:

162. Here is a modified version of our current gauss_random() routine. It is my expectation that we would use a substitute gauss-random ( ) compiled by you and *loaded with our code* to run your random numbers." "We would expect to test with *different numbers of paths*. Our program requires that the number of paths be REDACTED unless you have another suggestion. We are currently engaged in a project to determine accurate mean values for sample securities. Getting your results without those results would not be too helpful, so my only *urgency* is to be sure there is no great time disparity in finishing our sampling [with the large batches using thousands of seeds for Monte Carlo analysis] and testing with your random numbers."

E8. Defendants used ACE Theft Code in the last ACE test and stole all ACE sequences.

163. From the evolution of the ACE Theft Code files with progress of ACE tests, and the original records from the last test on June 3rd, 1998, it is clear that for the last ACE test, Russell compiled the REDACTED used to test ACE with the built-in Theft Code, and ran it in the ACE test system and stole all ACE sequences. This is consistent with Complaint ¶¶66-68, Ex. LLL.

164. Recall that corresponding to a char, say "A" following REDACTED REDACTED

165. In ACE test system (which is similar to the ACE Use Code in The Yield Book offline system in that both of them read in the ACE sequences verbatim), for example, CGM4655 indicated that, at 4:34pm, pass-throughs were tested for ACE64 (which is read in from the file REDACTED The application REDACTED

REDACTED

REDACTED

REDACTED Here I used the char 6 for ACE64 base scenario, u for up, d for down, s for steep, i for invert, t for 10% up. Then the function REDACTED REDA to save all 6 term structures corresponding to the 6 scenarios generated by ACE64.

166. After the 4th ACE test was completed, Defendants ran the ACE test system with ACE Theft Code again. The application REDACTED was launched with the same options REDACTED " for the example above. The function REDACTED REDACT

167. CGM4653 and CGM4655 showed ACE Theft Code in action. Pass-throughs are whole loans, and are the easiest and quickest to be priced. IO's and PO's are strips from pass-throughs, and much harder to price. CMO's may take even longer. During the 4th ACE test, for each ACE sequence, 20 passthroughs were run first, and then 30 IO's and 30 PO's were run. We would expect that the time to run 20 passthroughs was about 1/3 of the time for the 60 IO's and PO's. However, because for each ACE sequence, the function REDACTED RED is run to price pass-throughs, so the time for pricing 20 passthroughs was about the same the time for 60 IO's+PO's, and even mostly longer than the time for the 20 CMO's.

168. Defendants' claim that it takes less than a second to generate term structure. From the recorded time, we can estimate how much time it took them to save the ACE term structures. As CGM4655 indicated, at 4:34pm, ACE64 priced the 20 pass-throughs (running together with the function REDACTED;). It took 16 minutes 53 seconds to save the price output in the directory xiaolu REDACTED. Then at 5pm, ACE64 priced the 60 IOs, POs. It took 17 min 53 seconds To save the price output in the directory xiaolu REDACTED. So the REDACTED) took about approximate 17:53-17:22/3 =11 minutes to save term structure for ACE64.

169. To use ACE, this time consuming and elaborate REDACTED operation - is absolutely necessary because their online The Yield Book code cannot generate the ACE term structures, the term structures if REDACTED is used. When ACE is used, the program must 'REDACTED' the term structure 'REDACTED' by offline The Yield Book code. If 'REDACTED' is not REDACTED, meaning it is REDACTED, there is absolutely no need to spend 10 minutes to "save and reuse" the term structure that it is obtained in seconds and is already in the RAM.

E9. Defendants' admission that they and others would be motivated to steal ACE and use it, reflects exactly what they did to obtain the means to pursue an MBS Arbitrage Strategy

170. Defendants admitted that there was a "temptation" to "steal" ACE during Defendants onsite tests and the "main goal to remove the temptation for anyone to steal" ACE, after ACE was used by Defendants in production. These seem to be the motivation for the design of ACE Theft and Use Code to prevent ACE theft by third party. *See* the 9/17/96 email from Russell to Dr. Wang (CGM 00096) (Ex. PPP) which states:

> There are two concerns in our testing your numbers. The first is that we not steal them from you. The second is that no one steals them from us, either while testing or thereafter. Clearly one solution is that we never obtain the sequence from you and that you test our code at your site. Unfortunately this would take a lot of work (and concomitant delay) for us to get some sort of substitute data base to run our CMO's at your site. In addition, Salomon is paranoid about releasing any of its code, in whatever form.
>
> Lakhbir believes that we should do the testing here, as this is an absolutely necessary step to decide whether Salomon should purchase the sequence. We would provide you with suitable assurances of indemnity. Should the numbers be stolen (by us or third parties), at least they could be identified by the suggested markings. But I am still a bit concerned about security from third parties. Perhaps you are aware of security measures that we might employ to prevent someone here from taking the numbers and selling them.

171. Also see the 09/18/96 (CGM 00066)(Ex. QQQ) email from Russell to Defendants' system administrator Lifen Wang:

> We would like to be able to run some things so that no one else in the firm would have access to the code or the process." "The reason for this is to prevent theft of certain data *that has a high market value.* We need to test this data before purchasing it. After the purchase, we would still need high security to prevent its theft, but this problem can be delayed until after our testing is completed. *Our main goal is to remove the temptation for anyone to steal the data.*"

172. And finally, on 9/18/96 Russell emailed Dr. Wang (CGM0036)(Ex. RRR) advising him that they could get a separate machine ready for testing. It opines that

> [t]his approach to the testing appears to provide a very high degree of security. Unfortunately, the same approach probably cannot be used for production work. But we do not have to worry about that yet.

173. It appears that The Yield Book code withheld by Defendants used ACE surreptitiously. This is the same idea as separating the ACE test system from The Yield Book systems. Defendants seemed to be confident about this idea and to be willing to provide suitable assurances of indemnity, in order to persuade Dr. Wang to agree to do the testing on Defendants' site. Given the potential benefits of ACE facilitating their ability to engage for the first time in arbitrage of illiquid MBS, it is clear why security was such a big issue. I can see that this was an absolutely necessary step to steal ACE, Defendants seem to have tested Numerix's sequences on Numerix' site at about the same time as ACE first tests and showed no interests in further tests[52])

[52] Ex. TTT, September 17, 2007, Russell dep: 190:22-191:5 (Regarding an E-mail dated June 12, 1997, from Russell to Numerix. REDACTED

F. Defendants surreptitiously switch to ACE *right after* they stole ACE in the last ACE test in June 1998 and relied on ACE for their production until at least 2007

174. On June 16, 1998, less than two weeks after the theft of ACE on June 3, 1998, which was *right after* they backed out the ACE sequences, Defendants immediately installed the ACE Use Code. The ACE Use Code surreptitiously switched from the term structures (interest rate paths) generated by Defendants' Monte Carlo code[53] in The Yield Book to the term structures generated by ACE in The Yield Book production system for their traders. See the flowchart Fig.3 in Ex. SSS. Some of the new functions of ACE Use Code are shown in grey as shown the color flow chart, Ex. CCC. The Yield Book daily production started playing a shell game under the cover of their code files name literally: `mainshell.cc,v`. Ex. FFF, and `pathgen_cover.c,v`, Ex. EEE)(Underlining added). Under the flag "-x", the Offline ACE Use Code generated, "saved" ACE rate paths, and "stored" them in a hidden directory. Then under the flag "-x", The Yield Book running the Online ACE Use Code for their traders surreptitiously replaced the Monte Carlo rate paths generated by their random number generator by the ACE rate paths "retrieved" in the hidden directory for their production.

175. Defendants designed ACE Theft Code with the ACE Use Code in mind, so most of the ACE Theft Code became a part of ACE Use Code, as discussed in detail below (F1). The functions of ACE Theft Code in brown (Ex. CCC) were converted to ACE Use Code in grey, by lifting the restriction #ifndef YB on June 16, 1998, as shown in Ex. CCC and the flow charts Fig.2 and Fig.3 of Ex. SSS. They promptly removed the Theft Code that was no longer needed (F6). In addition, to build in sufficient robustness into the production code for use every day, they installed at least two completely new suites of code in the ACE Use Code (F2 and F3), one in Mortgage Research RCS in the term structure engine for The Yield Book (F2) and another in RCS code in The Yield Book, Inc. in chassis (F3) to put ACE into The Yield Book production.

176. The daily substitution of the ACE Interest Rate Paths for Defendants' paths for their **internal users** resulted in conflicts between the **online code** that retrieves the ACE Interest Rate Paths and the **offline code** which was putting in the updated ACE Interest Rate Path into the offline file, theREDACTED to be retrieved and substituted for Defendants' rate paths by the **online code**. Attempted retrieval of the ACE Interest Rate Paths from offline by the **online code** during updating would cause a conflict. So Defendants installed a software "lock" mechanism to resolve the conflicts (Section F5).

177. Defendants' hid the information about where the directory containing the file with the ACE Interest Rate Paths away from the employees with access to the **online** RCS computer code. However, this required those who were cooperating in the cover up to manually enter, via the keyboard or a script file, the directory and file path for the files containing the ACE Interest

[53] As explained by Defendants' head of Mortgage Research, Dr. Hayre, "Monte Carlo" analysis is the name of the type of analysis performed by The Yield Book which generates future interest rate paths and uses them to generate cash flow value MBS. They use Monte Carlo analysis because the prepayment option in residential MBS causes their cash flow to vary with interest rate movement. Hayre August 17, 2007 dep 51;11-53;25. Ex. WWWW. Hence, the interest rate paths are sometime referred to as Monte Carlo paths.

Rate Paths and the number of paths being used each time they started The Yield Book program. To remind the conspirator programmers what the code was supposed to do, Defendants buried coded comment[54] reminders in the source code. These reminder comments served as a guide to unravel their shell game.

F1 <u>Right after the ACE theft, Defendants started installing ACE Use Code in The Yield Book, this code surreptitiously replaced "Defendants' sequence" by ACE sequences in production</u>

178. I analyzed the versions of REDACTED right after the last ACE tests on June 3, 1998 through July 2005, the date when RCS code was produced in the Production March 2012. It alone shows that right after June 3, 1998, Defendants quickly put the stolen ACE sequences into TYB production replacing their Production200 sequence by replacing the ACE Theft Code with the ACE Use Code.

179. This is clear from the code. I compared the two versions of REDACTED, v1.21 of April 8, 1998, Ex. DDD, with V1.22 of July 9, 1998, Ex. VVV, right before and right after June 1998 theft. This comparison as shown by the RCS archive version, REDACTED (Ex. EEE), revealed that:

a. The ACE Theft Code V1.21 has login comment: REDACTED . Running REDACTED REDACTED and then stole ACE term structures.).

b. The ACE Use Code V1.22 REDACTED It appears that it was imported straight from a TYB-RCS production code file, as we shall see below. This suggested that the previous restriction REDACTED , (if not for Yield Book production) code on the REDACTED option had been altered to be REDACTED meaning now for Yield Book production. A Yield Book application using the REDACTED option now REDACTED REDACTED in order to add ACE term structures for production. The following are all of the differences between V1.21 and V.1.22 and they confirm this conclusion:

(1) V.1.22 (Ex. VVV REDACTED
REDACTED
REDACTED

REDACTED is the variable used by the Theft Code to hold the ACE Interest Rate Paths. So this comment is saying "get" the ACE Interest Rate Paths from REDACTED So now REDACTED is used for TYB production. Then v1.22 inserted the following code on the bottom of Page 1, L42:

[54] "Comments" are a standard feature in C++ and Fortran software language that allow programmers to write explanatory notes within the source code which is a text file that cannot be read by the computer. When the source code is compiled by a compiler program into executable code that can be read by the computer, the comments are ignored.

REDACTED

As before REDACTED stands for "not for Yield Book" but, rather, for Defendants' **inner circle** people who can run the **offline** ACE Use Code to generate term structures using the ACE sequences. So REDACTED means for TYB production, for the Yield Book users. This is consistent with the "REDACTED i.e., Yield Book, specifying the function REDACTED he Theft Code.

V1.22 (Ex. VVV) inserted the following code just before the end of the function REDACTED in L160, Page 3:

REDACTED

Therefore REDACTED for the **inner circle** to use REDACTED REDACTED for TYB production.

(3) V1.22 in L215, p. 4 (Ex. VVV) REDACTED

REDACTED

[55] REDACTED obviously means REDACTED , namely, not for Yield Book users. See Teytel August 23, 2013, dep. REDACTED REDACTED